The Synthesizability of Texture Examples

Dengxin Dai Hayko Riemenschneider Luc Van Gool
Computer Vision Lab, ETH Zurich, Switzerland
{dai, hayko, vangool}@vision.ee.ethz.ch

Abstract

Example-based texture synthesis (ETS) has been widely used to generate high quality textures of desired sizes from a small example. However, not all textures are equally well reproducible that way. We predict how synthesizable a particular texture is by ETS. We introduce a dataset ($21,302$ *textures) of which all images have been annotated in terms of their synthesizability. We design a set of texture features, such as 'textureness', homogeneity, repetitiveness, and irregularity, and train a predictor using these features on the data collection. This work is the first attempt to quantify this image property, and we find that texture synthesizability can be learned and predicted. We use this insight to trim images to parts that are more synthesizable. Also we suggest which texture synthesis method is best suited to synthesise a given texture. Our approach can be seen as 'winner-uses-all': picking one method among several alternatives, ending up with an overall superior ETS method. Such strategy could also be considered for other vision tasks: rather than building an even stronger method, choose from existing methods based on some simple preprocessing.*

1. Introduction

A substantial amount of work has been devoted to synthesising textures from examples [13, 33, 7, 16, 19, 27, 5]. We will refer to such example-based texture synthesis as 'ETS' in the remainder of this paper. Even if the set of textures that can be successfully synthesised that way has steadily been growing, it is often not clear beforehand whether ETS would be successful for a specific texture sample. It would be interesting if we were able to predict its synthesizability – how well its underlying visual patterns can be re-synthesized by learning only from the sample. Even if challenging, the task may be doable, given that other qualitative image characteristics could be quantified, like quality [26], memorability [14], or interestingness [11].

While ETS has proven to be a powerful tool to generate large-scale textures [37], providing texture examples is not straightforward [25]. ETS systems typically expect a rectangular sample image, representing a head-on view of a flat, outlier-free textured surface. Not just any example image returned by an image searching engine (by typing keywords) will do. Such retrieved images usually contain outliers, cluttered backgrounds, distorted texture surfaces, or even objects and complex scenes. Being able to rank retrieved images in terms of their synthesizability can then at least perform an initial selection. It can also be used to trim

Figure 1. Synthesizability of texture examples detected by our system. The values are in $[0, 1]$ and a higher value means the example is easier to synthesize. All images are of 300×300 pixels.

images to regions with good synthesizability, by e.g. removing undesirable background. Furthermore, synthesizability can help select an appropriate ETS method. The optimal approach – also taking into account speed and stability – will depend on the texture and the application. Quilting [7] is very potent, for instance, but will tend to produce verbatim repetitions that become salient when larger areas need to be synthesized. It would be good if in such case one could take recourse to an alternative method that does not produce such issues. Last but not least, studying synthesizability as a general image property is interesting *per se*.

Fig. 1 shows the synthesizability scores assigned to some texture samples by our system. Fig. 11 illustrates the trimming of an image to its most synthesizable, rectangular regions (the red cut-outs).

In order to learn image synthesizability and evaluate its performance, we have collected a fairly large texture dataset of $21,302$ texture images. This dataset has been manually annotated in terms of the synthesizability of each image. The synthesizability is characterised as the 'goodness' of the 'best' synthesis result as obtained by a set of ETS methods. The 'goodness' is quantified as one of three levels: good, acceptable, and bad. See Fig. 2 for examples.

As to the automated synthesizability scoring, a series of features that would seem to be connected with the task are defined. A scoring function is then learned from the collection of annotated data. The experimental results show that automated synthesizability scoring is possible.

Our main contribution are: (1) to learn the image property synthesizability methodologically; (2) to design several novel features for qualitative texture analysis (esp. 'textureness', homogeneity, repetitiveness, and irregularity); and (3) to offer a fairly large texture dataset together with synthesizability annotations;

The remainder of the paper is organized as follows. Sec. 2 reports related work. Sec. 3 is devoted to our dataset, followed by our features and learning method in Sec. 4. Sec. 5 presents our experiments and Sec. 6 concludes.

2. Related work

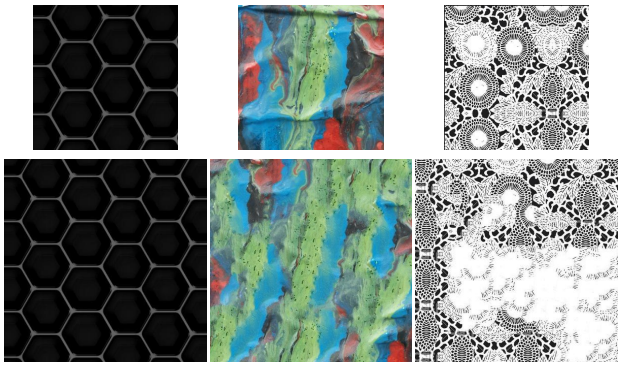
Example-based texture synthesis. Techniques of example-based texture synthesis can be broadly categorized into four categories: feature-oriented synthesis [13, 6, 33, 9], Markov Random Fields (MRFs) methods [32, 40, 39], neighborhood-based methods [7, 16, 15, 5], and tile-based methods [4, 23]. The first group learns the statistics of carefully designed features and leads the synthetic images to have/achieve similar values, e.g. color histograms [13], multi-band spatial frequencies [6], and wavelet features [33]. This group of methods are stable and do not generate verbatim repetition, but the main challenge lies in designing a common set of features that is able to capture the essence of all kinds of textures. The second group considers textures as instances of MRFs. Parameters of the MRFs are estimated from the texture examples and new textures are then sampled from the model. Multi-scale neighborhood-based MRFs are learned in [32] and pairwise clique-based MRFs in [39]. This strand is theoretically well-founded, but is computationally expensive. The third group generate textures by copying pixels or patches from the exemplar inputs [8, 7, 16, 15, 5]. Unlike the first two groups they do not provide cues for texture analysis, but are often more efficient and tend to work for a larger variety of textures. The last group assemble new textures out of a set of (rectangular) tiles cropped from example images. This stream of methods are very efficient once the tiles are estimated. However, identifying these tiles is non-trivial: [24] handled this problem by estimating the translation symmetries and [5] through semantic labeling.

Texture recognition. Our work is also related – albeit rather weakly – to material recognition. Features for material classification include statistics of filter responses [20, 28, 34], joint intensity distributions within a compact neighborhood [36, 22], geometric features over topographic maps [38], and high-level semantic attributes [29]. Similarly, we need to design appropriate features for this new task.

3. Data collection

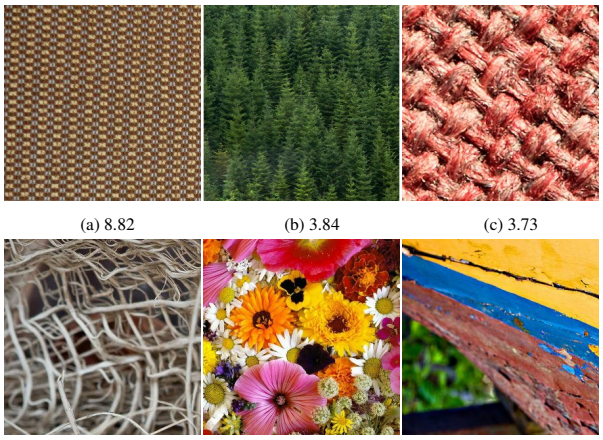
Although it stands to reason that some textures are easier to synthesise than others, quantifying this expectation has not been addressed. In order to learn to predict synthesizability, we collected a texture dataset and annotated it in terms of synthesizability. $40,000$ images were downloaded from Google, Bing, and Flickr by providing 60 keywords. The keywords used are to cover common material classes such as glass, water, stone, plastic, fabric, leather, metal and paper, and to cover common geometric texture attributes such as stochastic, repetitive, lined, speckled, wrinkled and cracked. All images were truncated to 300×300 pixels, with images smaller than this not being used. Since the retrieved images are very 'noisy', with some not showing textures, being of low quality, or being severely watermarked, we made a manual selection for the truncated images. Finally, we ended up with a dataset of $21,302$ texture samples.

For the annotation, we characterize the synthesizability of a texture as the 'goodness' of the best synthesized image among those generated by a selected set of ETS methods. A good synthesized image should be as similar as possible to the input example and should not have visible artifacts such as seams, blocks and ill-shaped edges, and should not contain salient repetitions of sub-patterns in a verbatim fashion, if that is not the case in the original. Since no single ETS method performs better than all others on all kinds of textures, the annotator got the choice between the results of four specific methods, that are based on different



(a) good, Quilting (b) acceptable, MMRF (c) bad, NULL

Figure 2. Three texture examples from our dataset with their annotations of synthesizability. Top: texture exemplars; bottom: synthesized textures.



(a) 8.82 (b) 3.84 (c) 3.73

(d) 3.12 (e) 1.80 (f) 1.63

Figure 3. The homogeneity of texture examples detected by our method. Images are all of 300×300 pixels.

methodologies: an image quilting method [7], a multi-scale Markov Random Field (MMRF) method [39], a wavelet-based parametric method [33], and a random phase synthesis [9]. While future work will probably yield more powerful ETS methods still, this dataset constitutes an initial benchmark, based on the current state-of-the-art in ETS. The final outcome of the annotation for a texture example is the 'goodness' of the synthesized result (among the 4) that an expert annotator considered best. This goodness was expressed as one of 3 levels: good, acceptable, and bad, assigned synthesizability scores of 1, 0.5 and 0, resp. The 'best' method of each texture example was also recorded to learn which method is the best to synthesize a given texture example. This was only performed for 'good' and 'acceptable' images; 'bad' ones were assigned to 'NULL'. Fig. 2 shows examples of such annotation. In total, 25.5% samples were labeled bad, 39.7% acceptable and 34.8% good.

4. Learning image synthesizability

In this section, we investigate the visual features relevant to image synthesizability. We start from general image features, to move on to our designed texture features, and to the learning method.

4.1. General features

Local patterns. Local binary patterns (LBP) [30] have been widely used in texture recognition and such features achieved s-o-a classification performance [22]. Thus, we included uniform LBP.

Filter responses. Using image filters has become one of the most popular tools for texture analysis [20, 28] and synthesis [40]. Thus, filter bank responses may be helpful for learning synthesizability too. The Schmid Filter Bank [34] is employed with 13 rotationally invariant filters at 5 scales.

GIST features. Frequency analysis has proven very useful for texture analysis/synthesis [10, 28, 6], so features of this kind can best be included here as well. GIST [31] is used, where the implementation resizes images to 256×256 pixels, only considers one grid, and produces a feature vector of dimension 20.

4.2. Designed features

'Textureness'. Objects and scenes are more difficult to synthesize than actual textures. We train a classifier to distinguish textures from objects and scenes. The UIUC texture dataset [17] delivered the positive samples (textures), and the 15-Scene dataset [18] the negative ones (objects/scenes). Linear SVMs were used as the classifier with GIST [31] as the feature. The classification score is taken as 'textureness'.

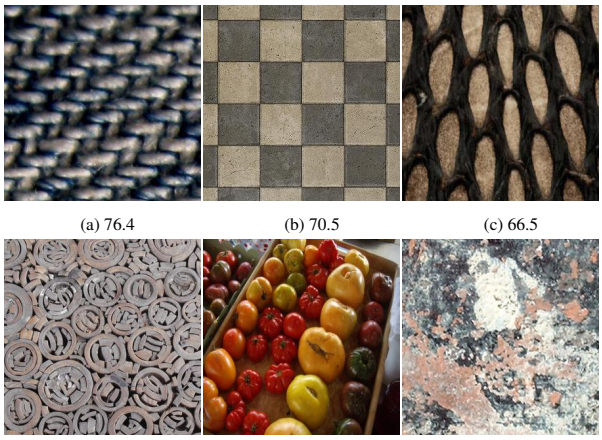
Homogeneity. Homogeneous textures are easier to synthesize than heterogeneous ones. Thus, it is desirable to have a feature measuring homogeneity. One possibility is based on co-occurrence matrices [35], but is low-level and quite noise sensitive. We here propose a simple, yet more robust method based on our definition of homogeneity.

Definition: The homogeneity of an image is the expectation of visual similarity between two randomly-chosen local regions of the image.

In particular, given an image $X \in \mathbb{R}^{H \times W}$, we measure the average similarity over T (80 in the implementation) trials. In trial t, two regions R_1^t and R_2^t are sampled from X, and their distance $d(R_1^t, R_2^t)$ is measured. The homogeneity of X is then:

$$\mathrm{Hom}(X) = \frac{1}{T} \sum_{t=1}^{T} \frac{1}{d(R_1^t, R_2^t)}. \qquad (1)$$

where $d(\cdot, \cdot)$ is the Euclidean distance, R_1^t and R_2^t are of the same size $\lfloor H/3 \rfloor \times \lfloor W/3 \rfloor$, and the positions of their top-left corners are sampled uniformly, at random from



(a) 76.4 (b) 70.5 (c) 66.5

(d) 49.7 (e) 39.4 (f) 32.4

Figure 4. The repetitiveness of texture examples detected by our method. Images are all of 300×300 pixels.

$\{(i,j) : i \in \{1,...,\lfloor 2H/3 \rfloor\}, j \in \{1,...,\lfloor 2W/3 \rfloor\}\}$. The regions are represented with bag-of-words. The dictionary is learned from X by k-means with 30 'word' centres and with 10×10 patches around every pixel (RGB values are used). See Fig. 3 for the homogeneity of six texture examples detected by the method. Our homogeneity is more effective than the co-occurrence one [35] because it uses regions rather than single pixels. It is also more robust because the word histograms yield some spatial invariance.

Repetitiveness. Textures are usually referred to as visual surfaces composed of repeating patterns, that are similar in appearance [37]. FFT features, of which the power spectrum is directly related to auto-correlation, have been used very early on [10, 35, 21]. For periodic patterns, the auto-correlation function is strongly peaked. Here we propose a related measure, also aimed at capturing imperfect repetitions (Fig.4), that is defined in the spatial domain.

The method draws on normalized cross correlation (NCC): an image $X \in \mathbb{R}^{H \times W}$ is cross-correlated with itself, generating an NCC matrix $D \in \mathbb{R}^{(2H-1)\times(2W-1)}$. The elements in the matrix are divided by the number of pixels involved in their calculation (different overlap as the image is shifted across itself). The borders of the matrix are not used due to the insufficient overlap there. The idea is that if X is repetitive, the following two properties should hold: (1) for a random moderate-sized region R of D, the difference between its maximum value and its minimum value should be large; (2) the minimum values of a set of randomly sampled R's (of the same size) should be very close. The philosophy behind (1) is that for repetitive textures, the auto-correlation function should exhibit peaks and valleys. Property (2) is derived from the fact that the distances between all 'repeated' versions should be similar.

Denoting by $Max(R_t)$ and $Min(R_t)$ the maximum and minimum values of the tth region R_t, we quantify the repetitiveness of X as

$$\text{Rep}(X) = \left(\frac{1}{T} \sum_{t=1}^{T} \frac{Max(R_t)}{Min(R_t)} \right) \times \frac{1}{\sigma(Min(R_t))} \quad (2)$$

where T (80 in the implementation) is the number of randomly sampled R's in D, and $\sigma(z)$ is the standard deviation of z. The size of R is set to $\lfloor H/5 \rfloor \times \lfloor W/5 \rfloor$. Too small a size cannot capture large-scale repetition, and too large a size looses discrimination power. See Fig. 4 for examples of detected repetitiveness. Repetitiveness is akin to the Harmonicity feature of [21], but repetitiveness is more robust due to its pooling over local regions.

Irregularity. Irregular textures are harder to synthesize than regular ones [23], so we conjecture that the irregularity of textures is also relevant to their synthesizability. Although the irregularity of textures has been suggested before, we still lack a method to measure it computationally. We propose Ensemble Composition (EC) for such quantification. The idea is that if a texture is regular, composing any of its regions using image chunks from outside will be *cheap* (See images in Fig. 5 to get the idea). We again do this over an ensemble – over T trials (80 in the implementation), we use the average composition energy to indicate texture irregularity. In the t-th trial, given an image X, we denote by R^t the region to compose, and by Y^t the rest of the image. The composition should have two properties: (1) the composited region should be similar to R^t; (2) the chunks from Y^t should be as continuous (large) as possible.

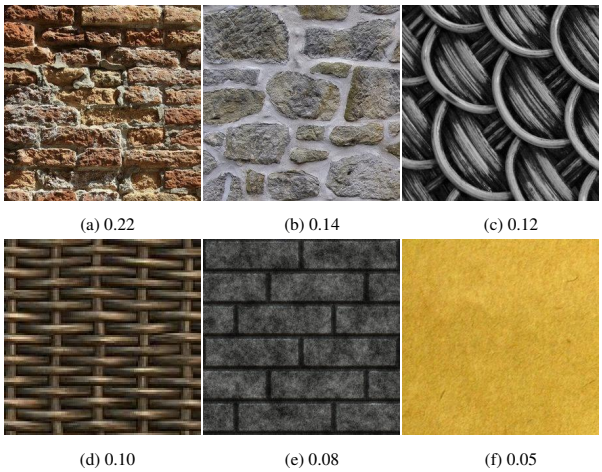
We formulate the composition task as a graph labeling problem with the following energy:

$$E(R^t) = \sum_{i \in R^t} D_i(l_i) + \lambda \sum_{\{i,j\} \in \mathcal{N}} V(l_i, l_j) \quad (3)$$

where l_i is the label assigned to pixel i in region R^t, and $\mathcal{N}$ is the neighborhood set of pixels in R^t. The label l_i represents the pre-defined offsets $\mathbf{s}_{l_i}$ between the composed pixels and composing pixels in the 2D image domain, that is, $l_i \in \{1,...,\#(X)\}$. $D_i(l_i)$ denotes the cost of assigning the l_ith label to the ith pixel of R^t, and it is defined to reflect the similarity of pixel i and corresponding shifted pixel $i + \mathbf{s}_{l_i}$. To counteract noise, we use the Euclidean distance between the Schmid Filter responses [34]; positive infinity is used when the shifted position falls outside of Y^t. For the smoothing term $V(l_i, l_j)$, we use the Potts model, i.e. $V(l_i, l_j) = 0$ if $l_i = l_j$ and 1 otherwise. λ is set to 50 to balance the two energy terms. By performing T trials, the irregularity of texture X is then defined as:

$$\text{IReg}(X) = \frac{1}{T} \sum_{t=1}^{T} E(R^t). \quad (4)$$

The energy is optimized by multi-label graph-cuts [2]. In order to speed the optimization up, we employed the tech-



(a) 0.22 (b) 0.14 (c) 0.12

(d) 0.10 (e) 0.08 (f) 0.05

Figure 5. The irregularity of texture examples captured by Ensemble Composition. Images are all of 300 $\times$ 300 pixels.

nique of dominant offsets proposed in [12] – only the dominant offsets (60 in our implementation) were considered as the labels. We also approximated the nearest neighbor search (for dominant offsets) by clustering patches into clusters (200 in our case) – patches in the same cluster are considered as neighbors. Our texture irregularity is similar to Boiman's image irregularity [1], but we focus on textures and compose the image from itself instead of composing general scenes from a dataset. Also, we provide an irregularity score for a given image by a new ensemble method.

It is noteworthy that homogeneity, repetitiveness, and regularity capture different properties. For instance, the texture in Fig. 3(b) is homogeneous, but not repetitive and not regular. The texture in Fig. 5(f) is homogeneous and regular, but not repetitive. In a nutshell, the 4 designed features are not orthogonal (*e.g.* repetitive textures are normally regular as well), but are complementary nonetheless. Moreover, we do not claim that the 7 features (general + designed) are optimal or exhaustive. Other features such as entropy, coarseness, directionality, could also be relevant to synthesizability.

4.3. Learning method

We attempt to computationally quantify texture synthesizability and to use this to aid synthesis. To those ends, we train (1) a regression model on the synthesizability scores (1, 0.5 and 0) to predict the synthesizability of a given image, and (2) an additional classifier to suggest the 'best' ETS method to synthesize it. Random Forest [3] was used for both training tasks due to its fast speed. 30 trees were used for the forest.

5. Experiments

5.1. Learning synthesizability

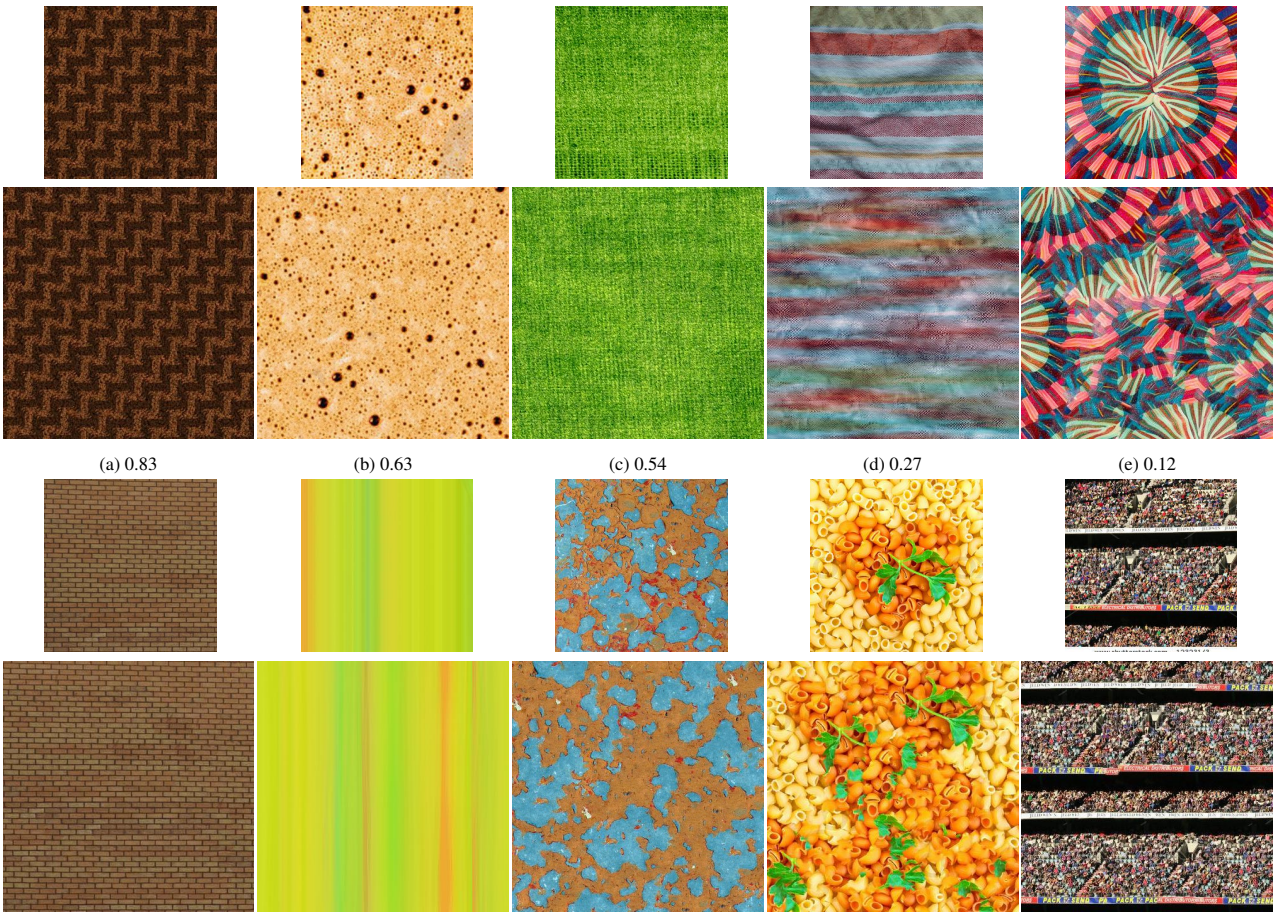
In this section, we evaluate the contribution of all features to the prediction of synthesizability and to what degree it is learnable. All 7 single features and their 3 combinations were evaluated. The 3 combinations are: combination of the 3 general features (General), combination of the 4 designed features (Designed), and combination of all features (All). 30% of the dataset images were used for training, the rest for testing. We report results over 5 random training-testing splits. For evaluation, we performed two retrieval tasks and evaluated the average precision for different levels of recall: (1) retrieve images with 'good' scores (>=good); (2) retrieve images with 'good' or 'acceptable' scores (>=acceptable).

Quantitative evaluation. Table 1 shows the results for different, single as well as combined features when recall is set to 1, and Fig. 8 shows the results for different levels of recall when all features are used. The table shows that every single feature is helpful. Homogeneity performs the best. It is also interesting that the combination of the 4 designed features performs substantially better than the combination of the 3 general texture features. This suggests that the designed features are indeed particularly relevant to synthesizability. This said, general texture features add to the power of the mix, given that the combination of all features yields the best performance. Also, from the highest precision scores (94.5% for >=acceptable, and 75.5% for >=good) we can conclude that image synthesizability is learnable and predictable. If only a fraction of well-synthesizable textures need to be retrieved, a very high precision can be expected (See Fig. 8). This is very useful for choosing synthesizable textures from internet images.

Qualitative evaluation. Fig. 1 and Fig. 6 show examples together with their predicted synthesizability. The synthesizability predictor here was trained with all annotated images except for the image itself given for prediction. As can be seen, homogeneous, repetitive, and regular texture examples obtain higher scores. The low scores are caused by many factors, such as outliers, surface distortions, and complex structures. In Fig. 6, the 'best' synthesised images by the ETS methods are also shown. The quality of the synthesized images is largely consistent with the predicted synthesizability score. This is crucial because it allows us to detect textures – also as image parts, see shortly – that can be synthesized well. As already claimed, the system can also suggest the 'best' synthesis method for a given texture example. Fig. 7 shows two such examples, where results of the suggested method and results of a randomly chosen method are compared. It can be seen that our 'adaptive selection' is superior to random guessing. This is due to the fact the ETS methods all have their own philosophies

Features	LBP	SFilter	GIST	Textureness	Homogeneity	Repetitiveness	Irregularity	General	Designed	All
>=Acceptable	88.1%	76.8%	84.1%	77.2%	88.7%	82.2%	76.7%	88.5%	93.1%	94.5%
>=Good	57.0%	37.8%	52.9%	38.5%	62.8%	45.6%	40.0%	60.2%	73.4%	75.5%

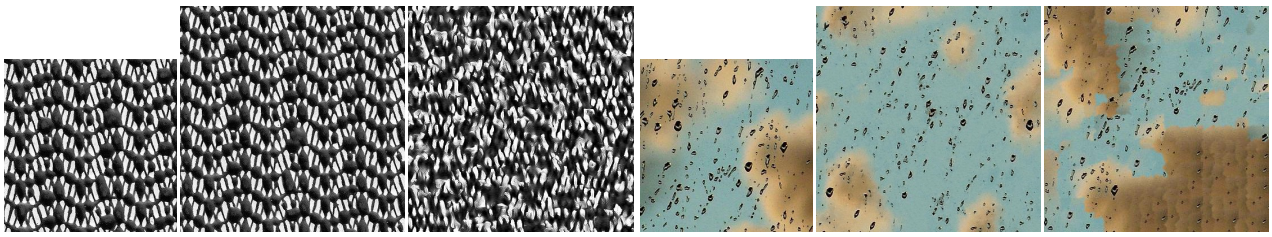
Table 1. The average precision of synthesizability prediction with all individual features and as combinations, when recall is 1.



(a) 0.83 (b) 0.63 (c) 0.54 (d) 0.27 (e) 0.12

(f) 0.74 (g) 0.68 (h) 0.49 (i) 0.23 (j) 0.19

Figure 6. Synthesizability scores of texture examples and the 'best' synthesized textures by ETS methods. Top: exemplar; bottom: synthesized.



(a) exemplar (b) suggested (Quilting [7]) (c) random (Wavelet [33]) (d) exemplar (e) suggested (MMRF [32]) (f) random (Quilting [7])

Figure 7. The synthesized results of two texture examples by our suggested method and a randomly chosen method.

and each one works better than the others for some textures, which necessariate an adaptive selection for the 'best' synthesis methods for a given texture example.

Failure cases. Of course, the method fails sometimes. The typical false positives (a high synthesizability score assigned to an image hard to synthesize) are images with fine, global, but irregular structures, *e.g.* crumpled paper and fabric, wood with year rings, foliage nerves, or hairs. It is hard to for the features to capture these subtle, but semantically crucial information. The typical false negatives (low score for synthesizable images) are heterogeneous textures such as some rust and cloud examples. This is probably because

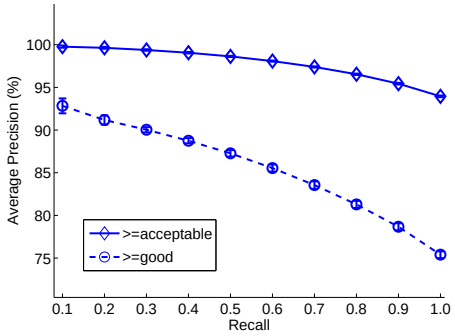
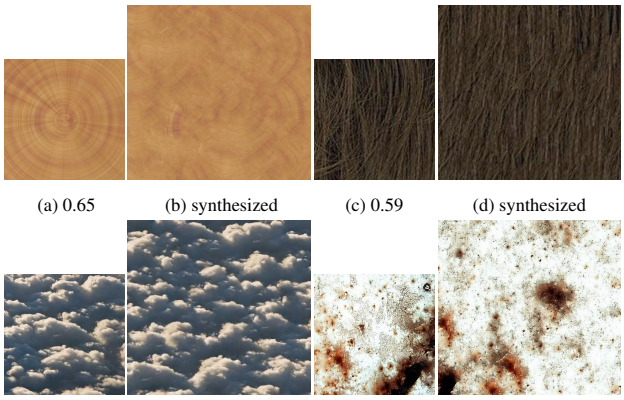


Figure 8. The average precision of synthesizability prediction for different levels of recall, when all features are used.



(a) 0.65 (b) synthesized (c) 0.59 (d) synthesized

(e) 0.35 (f) synthesized (g) 0.25 (h) synthesized

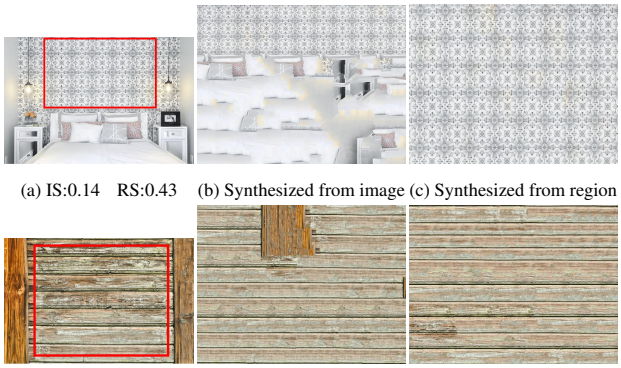
Figure 9. Failure cases: the top shows false positives and the bottom false negatives. Exemplars are of 300×300 pixels.

(a) 0.22 (b) 0.34 (c) 0.43

Figure 10. Synthesizability for different scales of the same texture.

the space of valid textures for those is very large so that synthesized textures more easily fall inside the space. See Fig. 9 for such examples.

Synthesizability with scales Textures in an image can be perceived and may differ at different scales. Thus, it is interesting to see how scales of textures affects their synthesizability. Fig. 10 shows an example, where three scales of the same texture are used as the examples for synthesizability prediction. Zooming in, the synthesizability score drops – as long as the same 'textons' matter – as those textons increasingly take on the role of individual objects. This is in keeping with human intuition.



(a) IS:0.14 RS:0.43 (b) Synthesized from image (c) Synthesized from region

(d) IS:0.13 RS:0.50 (e) Synthesized from image (f) Synthesized from region

Figure 12. Synthesis results for the images on the left, starting from the entire image (IS; mid column) vs. from the selected region (RS; right column).

5.2. Trimming texture examples

In this section, image synthesizability is used for trimming images to more synthesizable parts. Given an image, the synthesizability of subimages is computed and compared. The most synthesizable subimage is then suggested. See Fig. 11 for examples. 500 sub-windows were randomly sampled, with a minimum size of 100×100 pixels and maximum size that of the entire image. The figure suggests that our method performs well for this task. It is thus possible to pick synthesizable texture examples from unconstrained images. Fig.12 illustrates that synthesis is superior for the selected windows compared to using the entire image. Note that if two windows receive the same/close synthesizability score, we prefer the larger window.

6. Conclusions

The paper proposed synthesizability as a novel texture property and developed a computational predictor for it. We constructed a fairly large texture dataset and calibrated it according to synthesizability. A set of texture features have been proposed and, in some cases, designed for the learning, such as 'textureness', homogeneity, repetitiveness, irregularity. Extensive experiments show that image synthesizability can be learned and predicted computationally. It can be used to find good texture examples for synthesis, to detect good textures from unconstrained images for synthesis, and to choose an appropriate method to do so.

Our approach can be seen as kind of a 'winner-uses-all' strategy. Rather than aiming for the next best method, the idea is to rather pick one case-optimal method among several existing alternatives, with the goal of reaching success rates better than those of any individual method. Such eclectic strategy could also be tried for other tasks: rather than creating ever stronger methods, choose from existing methods based on some preprocessing.

(a) RS:0.50 IS:0.19 (b) RS:0.40, IS:0.22 (c) RS:0.53, IS:0.40 (d) RS:0.51 IS:0.23

Figure 11. The most synthesizable region as detected. The synthesizability of the whole images (IS) vs. the selected region (RS) are given.

Reproducibility. The Code, Data and more examples are available at: `www.vision.ee.ethz.ch/~daid/synthesizability`.

Acknowledgement. The work is supported by the ERC Advanced Grant VarCity and the ETH-Singapore project Future Cities.

References

[1] O. Boiman and M. Irani. Detecting irregularities in images and in video. *IJCV*, 74(1):17–31, 2007. 5

[2] Y. Boykov, O. Veksler, and R. Zabih. Fast approximate energy minimization via graph cuts. *PAMI*, 23(11):1222–1239, 2001. 4

[3] L. Breiman. Random forest. *MACH LEARN*, 45(1):5–32, 2001. 5

[4] M. F. Cohen, J. Shade, S. Hiller, and O. Deussen. Wang tiles for image and texture generation. In *SIGGRAPH*, 2003. 2

[5] D. Dai, H. Riemenschneider, G. Schmitt, and L. Van Gool. Example-based facade texture synthesis. In *ICCV*, 2013. 1, 2

[6] J. S. De Bonet. Multiresolution sampling procedure for analysis and synthesis of texture images. In *SIGGRAPH*, 1997. 2, 3

[7] A. A. Efros and W. T. Freeman. Image quilting for texture synthesis and transfer. In *SIGGRAPH*, 2001. 1, 2, 6

[8] A. A. Efros and T. K. Leung. Texture synthesis by non-parametric sampling. In *ICCV*, 1999. 2

[9] B. Galerne, Y. Gousseau, and J. M. Morel. Random phase textures: Theory and synthesis. *IEEE Trans. Image Process*, 20(1):257–267, 2011. 2, 3

[10] L. V. Gool, P. Dewaele, and A. Oosterlinck. Texture analysis anno 1983. *COMPUT VISION GRAPH*, 29(3):336 – 357, 1985. 3, 4

[11] M. Gygli, H. Grabner, H. Riemenschneider, F. Nater, and L. Van Gool. The interestingness of images. *ICCV*, 2013. 1

[12] K. He and J. Sun. Statistics of patch offsets for image completion. In *ECCV*, 2012. 5

[13] D. J. Heeger and J. R. Bergen. Pyramid-based texture analysis/synthesis. In *SIGGRAPH*, 1995. 1, 2

[14] P. Isola, J. Xiao, A. Torralba, and A. Oliva. What makes an image memorable? In *CVPR*, 2011. 1

[15] V. Kwatra, I. Essa, A. Bobick, and N. Kwatra. Texture optimization for example-based synthesis. *ACM Trans. Graph.*, 24(3), 2005. 2

[16] V. Kwatra, A. Schödl, I. Essa, G. Turk, and A. Bobick. Graphcut textures: image and video synthesis using graph cuts. In *SIGGRAPH*, 2003. 1, 2

[17] S. Lazebnik, C. Schmid, and J. Ponce. A sparse texture representation using local affine regions. *PAMI*, 27(8):1265–1278, 2005. 3

[18] S. Lazebnik, C. Schmid, and J. Ponce. Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories. In *CVPR*, 2006. 3

[19] S. Lefebvre and H. Hoppe. Parallel controllable texture synthesis. In *SIGGRAPH*, 2005. 1

[20] T. Leung and J. Malik. Representing and recognizing the visual appearance of materials using three-dimensional textons. *IJCV*, 43(1):29–44, 2001. 2, 3

[21] F. Liu and R. Picard. Periodicity, directionality, and randomness: Wold features for image modeling and retrieval. *PAMI*, 18(7):722–733, 1996. 4

[22] L. Liu, P. Fieguth, G. Kuang, and H. Zha. Sorted random projections for robust texture classification. In *ICCV*, 2011. 2, 3

[23] Y. Liu, W.-C. Lin, and J. Hays. Near-regular texture analysis and manipulation. In *SIGGRAPH*, 2004. 2, 4

[24] Y. Liu, Y. Tsin, and W.-C. Lin. The promise and perils of near-regular texture. *IJCV*, 62(1-2):145–159, 2005. 2

[25] Y. D. Lockerman, S. Xue, J. Dorsey, and H. Rushmeier. Creating texture exemplars from unconstrained images. *Tech. Report*, (TR1483), October 2013. 1

[26] Y. Luo and X. Tang. Photo and video quality evaluation: Focusing on the subject. In *ECCV*. 2008. 1

[27] C. Ma, L.-Y. Wei, and X. Tong. Discrete element textures. In *SIGGRAPH*, 2011. 1

[28] B. S. Manjunath and W. Ma. Texture features for browsing and retrieval of image data. *PAMI*, 18(8):837–42, 1996. 2, 3

[29] T. Matthews, M. S. Nixon, and M. Niranjan. Enriching texture analysis with semantic data. In *CVPR*, 2013. 2

[30] T. Ojala, M. Pietikainen, and T. Maenpaa. Multiresolution gray-scale and rotation invariant texture classification with local binary patterns. *PAMI*, 24(7):971–987, 2002. 3

[31] A. Oliva and A. Torralba. Modeling the shape of the scene: A holistic representation of the spatial envelope. *IJCV*, 42(3):145–175, 2001. 3

[32] R. Paget and I. Longstaff. Texture synthesis via a noncausal nonparametric multiscale markov random field. *IEEE Trans. Image Processing*, 7(6):925–931, 1998. 2, 6

[33] J. Portilla and E. P. Simoncelli. A parametric texture model based on joint statistics of complex wavelet coefficients. *IJCV*, 40(1):49–70, 2000. 1, 2, 3, 6

[34] C. Schmid. Constructing models for content-based image retrieval. In *CVPR*, 2001. 2, 3, 4

[35] M. Tuceryan and A. K. Jain. Texture analysis. In *Handbook of Pattern Recognition and Computer Vision (2nd Ed.)*, pages 235–276, 1993. 3, 4

[36] M. Varma and A. Zisserman. A statistical approach to material classification using image patch exemplars. *PAMI*, 31(11):2032–2047, 2009. 2

[37] L.-Y. Wei, S. Lefebvre, V. Kwatra, and G. Turk. State of the art in example-based texture synthesis. In *EG-STAR*, 2009. 1, 4

[38] G.-S. Xia, J. Delon, and Y. Gousseau. Shape-based invariant texture indexing. *IJCV*, 88(3):382–403, 2010. 2

[39] A. Zalesny, V. Ferrari, G. Caenen, and L. Van Gool. Composite texture synthesis. *IJCV*, 62(1-2), 2005. 2, 3

[40] S. C. Zhu, Y. Wu, and D. Mumford. Filters, random fields and maximum entropy (frame): Towards a unified theory for texture modeling. *IJCV*, 27(2):107–126, 1998. 2, 3